LOAN DOCUMENT

PHOTOGRAPH THIS SHEET

DTIC ACCESSION NUMBER

LEVEL

INVENTORY

D

AFRL-ML-TY-TP-2001-0014

DOCUMENT IDENTIFICATION

2000

DISTRIBUTION STATEMENT A
Approved for Public Release
Distribution Unlimited

DISTRIBUTION STATEMENT

HANDLE WITH CARE

ACCESSION FOR

NTIS	GRA&I	☑
DTIC	TRAC	☐
UNANNOUNCED		☐
JUSTIFICATION		

BY

DISTRIBUTION/

AVAILABILITY CODES

DISTRIBUTION	AVAILABILITY AND/OR SPECIAL	
A-1		

DISTRIBUTION STAMP

DATE ACCESSIONED

DATE RETURNED

DTIC QUALITY INSPECTED 4

20010214 027

DATE RECEIVED IN DTIC

REGISTERED OR CERTIFIED NUMBER

PHOTOGRAPH THIS SHEET AND RETURN TO DTIC-FDAC

DTIC FORM 70A JUN 90

DOCUMENT PROCESSING SHEET

PREVIOUS EDITIONS MAY BE USED UNTIL STOCK IS EXHAUSTED.

LOAN DOCUMENT

AFRL-ML-TY-TP-2001-0014

Fuzzy Control for Autonomous Ground Vehicles

A Technical Paper presented to the American Nuclear Society, Ninth International Topical Meeting on Robotics and Remote Systems, Seattle, WA, March 4-8, 2001

by

Jeffrey S. Wit, Carl D. Crane and David G. Armstong II

Wintec, Inc.
429 S. Tyndall Pkwy, Suite S
Panama City FL 32404

University of Florida, Center for Intelligent Machines & Robotics
Gainesville FL 32611

AIR FORCE RESEARCH LABORATORY
MATERIALS & MANUFACTURING DIRECTORATE
AIR EXPEDITIONARY FORCES TECHNOLOGIES DIVISION
139 BARNES DRIVE, STE 2
TYNDALL AFB FL 32403-5323

NOTICES

USING GOVERNMENT DRAWINGS, SPECIFICATIONS, OR OTHER DATA INCLUDED IN THIS DOCUMENT FOR ANY PURPOSE OTHER THAN GOVERNMENT PROCUREMENT DOES NOT IN ANY WAY OBLIGATE THE US GOVERNMENT. THE FACT THAT THE GOVERNMENT FORMULATED OR SUPPLIED THE DRAWINGS, SPECIFICATIONS, OR OTHER DATA DOES NOT LICENSE THE HOLDER OR ANY OTHER PERSON OR CORPORATION; OR CONVEY ANY RIGHTS OR PERMISSION TO MANUFACTURE, USE, OR SELL ANY PATENTED INVENTION THAT MAY RELATE TO THEM.

THIS TECHNICAL PAPER HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION AND IT IS AVAILABLE FROM THE AMERICAN NUCLEAR SOCIETY, 9TH INTERNATIONAL TOPICAL MEETING ON ROBOTICS AND REMOTE SYSTEMS, SEATLE, WA, MARCH 4-8, 2001

DAVID E. SHAHADY, Captain, USAF
Lead, Robotics Research Group

for

ALLEN D. NEASE
Chief, Force Protection Branch

RANDY L. GROSS, Col, USAF, BSC
Chief, Air Expeditionary Forces Technologies Division

FUZZY CONTROL FOR AUTONOMOUS GROUND VEHICLES

Jeffrey S. Wit
WINTEC, Inc.
429 S. Tyndall Pkwy, Suite S
Panama City, FL 32404
jeff_wit@wintec-inc.com
(850) 913-0717

Carl D. Crane III and David G. Armstrong II
Center for Intelligent Machines and Robotics
University of Florida
Gainesville, FL 32611
ccrane@ufl.edu, dga@cimar.me.ufl.edu
(352) 392-0814

ABSTRACT

The Center for Intelligent Machines and Robotics (CIMAR) at the University of Florida has worked in the area of autonomous ground vehicles (AGVs) for several years under the sponsorship of the Air Force Research Laboratory at Tyndall Air Force Base, Florida. The objective of the work is to develop technological capabilities that can be applied to a variety of Air Force needs and application areas. Recently, one of these capabilities required the design of a modular architecture for autonomous vehicle navigation. This new architecture, which is currently under development, is called Modular Architecture eXperimental (MAX). One of the unique features of this architecture is a generic message for controlling the motion of any autonomous vehicle. This paper describes a control technique, which uses this generic message, for navigating various autonomous ground vehicles.

The resulting technique uses two fuzzy model reference learning controllers (FMRLCs). One FMRLC controls the vehicle linear velocity, and the other controls the vehicle angular velocity. Both controllers are designed from parameters that are defined in the MAX interface document. They have been implemented and tested successfully on three different vehicles, a Kawasaki Mule with Ackerman steering, a K2A robot with three wheel synchronous drive, and a tracked vehicle called All-purpose Remote Transport System (ARTS).

1. INTRODUCTION

The Center for Intelligent Machines and Robotics (CIMAR) began working with autonomous ground vehicles (AGVs) in 1990 and has continued working with them to the present day. The focus of this work has been on areas required for automated navigation. These areas include path planning, determining vehicle position and velocity, path execution, and obstacle detection and avoidance. In 1991, CIMAR completed its first design and implementation of a vehicle capable of automated navigation. A Kawasaki MULE 500 all-terrain vehicle was modified for computer control and currently serves as a Navigation Test Vehicle (NTV) at the University of Florida. A path planner using an A* search algorithm was implemented to determine the shortest, obstacle-free path to a goal point.[1] An integrated inertial navigation unit (INU) and differential global positioning system (DGPS) provides real-time vehicle position and velocity feedback data.[2] Path execution was accomplished by mounting motors and encoders on the vehicle's steering wheel, throttle, brake and transmission, and by using PID controllers to attain a desired response. Finally, an array of sonar sensors, mounted on

the front of the vehicle, detects any unexpected obstacle in the vehicle's path. The NTV has undergone several revisions, over the years, as current technology and research continues to advance. Figure 1 shows a picture of the NTV as it is today.

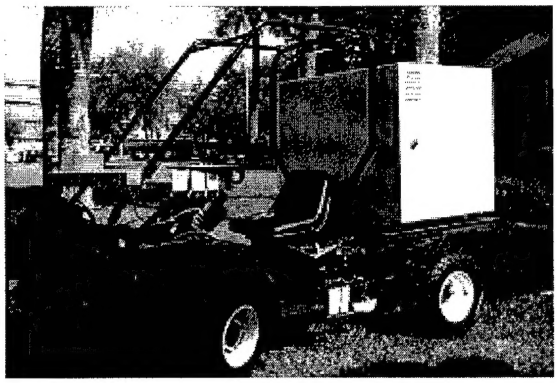

Figure 1: Navigation Test Vehicle (NTV)

In addition to the research in the above areas required for navigation, CIMAR has also concentrated its efforts in the development of an architecture for AGVs. The main requirement specified for this architecture was that it must allow systems to be comprised of self-contained submodules, where only the interface of each submodule is defined rigorously. The effect of this requirement benefits both the developer and the user. The developer now has a great amount of freedom in choosing specific hardware and software for his or her system. And, the user now has the ability to scale his or her AGV's functionality by combining different submodules. Developing an architecture that meets this requirement was accomplished with a two-step process by first determining a list of submodules required to automate a vehicle and then determining their interface. The Modular Architecture eXperimental (MAX), currently being developed at the University of Florida, attempts to meet this requirement.[3]

MAX currently consists of the following submodules: Position System (POS), Vehicle Control Unit (VCU), Path Planner (PLN), Detection and Mapping System (DMS) and Mobility Control Unit (MCU). Note that the POS, VCU, PLN, and DMS submodules coincide with the four areas mentioned earlier that are required for autonomous navigation. The MCU submodule is used to tie these four submodules together into one system. The modular structure of MAX is shown in Figure 2. The interface between each submodule defined by MAX allows communication with other submodules and/or the user.

The main task of the MCU is to control the mobility of the vehicle, which is done here by executing a planned path. A flow chart of the MCU and the submodules it utilizes is depicted in Figure 3, where the MCU begins by receiving a planned path from PLN. Once the MCU has this path, it uses position and velocity feedback data from POS and obstacle data from DMS to determine the controlled input to the VCU. In the work done here, it is assumed that the environment is known and static. Therefore, only the position and velocity feedback is used to determine the controlled input to the VCU.

A unique feature of the VCU is that the input message to control vehicle motion is generic for all vehicles. It uses two wrench commands, a propulsive wrench and a resistive

wrench, to produce and resist vehicle motion, respectively. Each wrench is comprised of a force vector, $\underline{f} = [f_x, f_y, f_z]$, and moment vector $\underline{m} = [m_x, m_y, m_z]$. The values for these two wrench commands are determined by first using a path tracking technique based on screw theory[4] called vector pursuit[5], which calculates the desired linear and angular velocities of the vehicle in order to follow the planned path. Then, two adaptive fuzzy controllers are used to track these desired velocities. The rest of this paper focuses on the development of these controllers. Section 2 gives a brief overview of the fuzzy controllers used. Sections 3 and 4 present the development of the controllers for the vehicle's linear and angular velocities, respectively. Section 5 shows the results of using these controllers on three different vehicles, a steered-wheeled vehicle, a tracked vehicle, and a 3-wheel synchronous drive vehicle. Finally, some conclusions are presented in section 6.

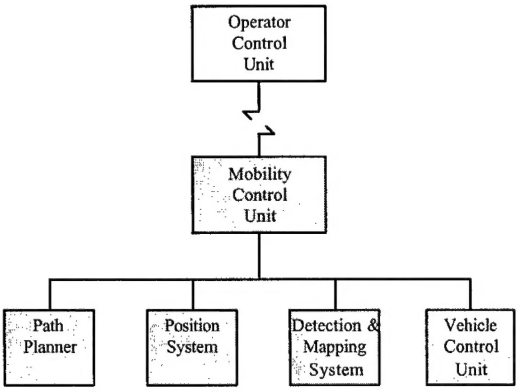

Figure 2: MAX submodule structure

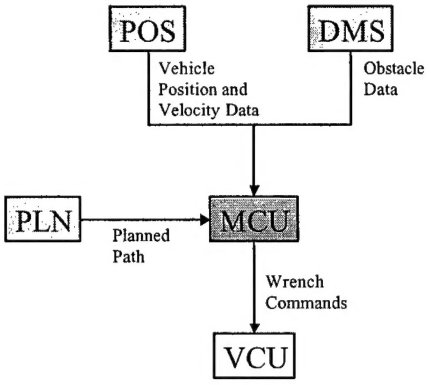

Figure 3: MCU flow chart

2. THEORETICAL BACKGROUND

There are two general techniques for adaptive control, direct and indirect. Direct adaptive control monitors a system's response and then modifies the controller in order to achieve a specified desired performance. On the other hand, indirect adaptive control monitors a system's response in order to identify parameters of the system's model. The controller is designed as a function of these model parameters to achieve a specified desired performance. The controller used here is called fuzzy model reference learning controller (FMRLC)[6], which is a direct adaptive controller. A block diagram of this controller is shown if Figure 4. This section is intended to give a brief summary of the FMRLC.

The main parts of a FMRLC are the fuzzy controller, the plant, reference model, and the learning mechanism. A fuzzy controller typically involves three steps: fuzzification, inference, and defuzzification. The fuzzification step takes the crisp inputs of the process and converts them to linguistic variables. The inference step uses these linguistic variables to decide the best course of action based on the knowledge of an expert, which is stored in a rule-base made up of a set of if-then statements. The defuzzification step takes the linguistic results of the inference step and converts them to crisp outputs. These crisp outputs of the controller are inputs to the plant, which is simply the system to be controlled. The reference model gives the desired system response based on the current input. The main constraint on the reference model is that it must

be reasonable. It is not reasonable to expect a system to achieve a better performance than what the system is capable. Every system has its limitations, and these limitations must be considered when choosing the reference model. Finally, the learning mechanism uses the outputs of the plant and of the reference model in order to calculate an error between the desired and actual response. This error is used then to decide how to modify the rule-base of the fuzzy controller in order to drive the error to zero.

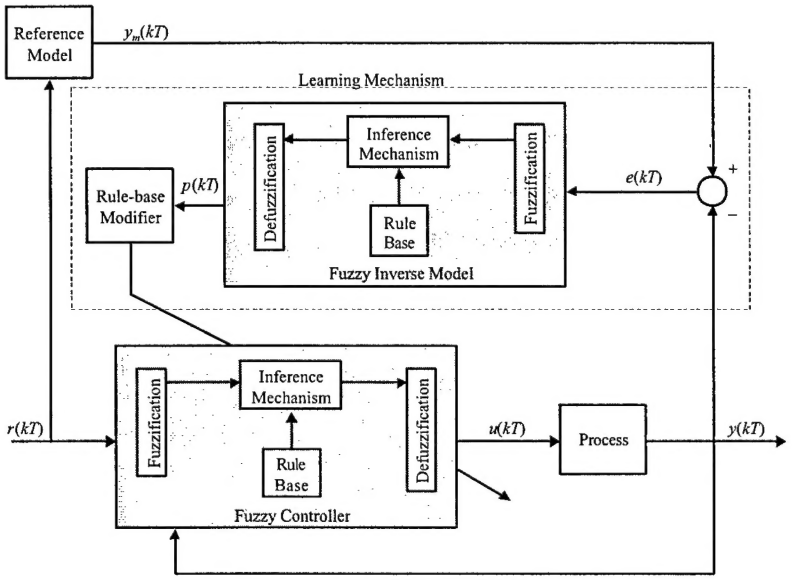

Figure 4: FMRLC Block Diagram

Once the reference model is determined, a discrete error signal can be calculated by:

$$e(kT) = y_m(kT) - y(kT), \tag{1}$$

where $e(kT)$ is the current error, $y_m(kT)$ is the output of the reference model, $y(kT)$ is the output of the system, and T is the sample time. Depending on the system characteristics, it may also be useful to calculate the discrete change in error by:

$$c(kT) = \frac{e(kT) - e(kT - T)}{T}, \tag{2}$$

where $c(kT)$ is the change in error, $e(kT)$ is the current error from equation (1), and $e(kT\text{-}T)$ is the error calculated on the previous time sample. Then, these results and any other system data are used to determine the necessary changes to the process inputs, $p(kT)$, by the learning mechanism.

The learning mechanism is made up of a fuzzy inverse model and a rule-base modifier. The purpose of the fuzzy inverse model is to take the calculations $e(kT)$ and $c(kT)$ and determine how to change the process input, $u(kT)$, in order to drive $e(kT)$ to zero. The output of the fuzzy inverse model is the desired change in process input and is represented by $p(kT)$. First, the inputs are fuzzified by membership functions specified by the designer. The inference mechanism then

uses rules such as, if the error is "positive small" and the change in error is "zero," then the change in process input is "negative small." It is referred to as the fuzzy inverse model because these rules typically depend on the plant dynamics. Finally, the output, $p(kT)$, is defuzzified by the center of gravity (COG), center-average or some other defuzzification technique. Then the output, $p(kT)$, is used to modify the controllers rule-base.

In order to modify the fuzzy controller's rule-base, which rules are active must first be determined. In other words, determine which rule's certainty is greater than zero:

$$\mu_{premise_{(i)}} > 0 . \tag{3}$$

Then, for all the rules that are active, the center of the m^{th} output membership function is adjusted by:

$$b_m(kT) = b_m(kT - T) + p(kT) , \tag{4}$$

where $b_m(kT)$ is the current center of the m^{th} output membership function, $b_m(kT\text{-}T)$ is the center of the m^{th} output membership function at the previous time sample, and $p(kT)$ is the desired change in process input that was calculated by the inverse model.

3. LINEAR VELOCITY FMRLC

The first task in designing a controller is to determine its inputs and outputs. Recall that under the MAX architecture, the propulsive and resistive wrenches are used to control the AGV's motion. Each wrench is made up of a force vector, $\underline{f} = [f_x, f_y, f_z]$, and a moment vector, $\underline{m} = [m_x, m_y, m_z]$. The propulsive wrench is used to propel the AGV in the direction of the force or about the axis of the moment. Since, by the careful selection of the vehicle's reference frame, the only term that has an affect on the linear velocity is f_x, it is chosen to be the linear velocity's controller output. One of the inputs to the controller is obviously the desired linear velocity, $v_{x,d}$. A second input to the controller is the vehicle pitch, θ_y, since it can have a substantial effect on the AGV's linear velocity. A block diagram of the FMRLC for the linear velocity is given in Figure 5.

From Figure 5, the controller's input $v_{x,d}(kT)$ is the desired linear velocity, and the controller's input $\theta_y(kT)$ is the vehicle's pitch. The gains, g_v and g_θ, are used to normalize the inputs. By doing this, both inputs are fuzzified using the membership functions given in Figure 6. Therefore, the gain g_v is chosen to be $1/v_{max}$, where v_{max} is the maximum velocity of the AGV, and the gain g_θ is chosen to be $1/\theta_{y,max}$, where $\theta_{y,max}$ is the maximum allowable pitch. Both of these terms, the maximum velocity and the maximum allowable pitch, are available from the VCU configuration message under the MAX architecture.

The controller's output $f_x(kT)$, in Figure 5, is the first term in the propulsive wrench. Using the output membership functions shown in Figure 7, the output of the inference mechanism is normalized also. The gain g_f is used to scale this output to allow the controller to command the entire range of the term f_x. In the MAX architecture, the term f_x has the range from –100 to 100 percent, and therefore the gain g_f is chosen to be 100.

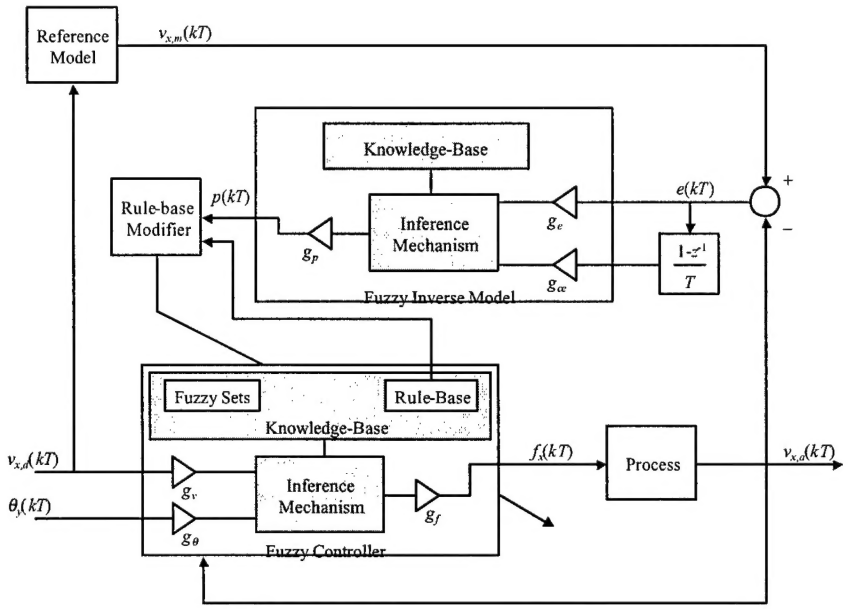

Figure 5: Discrete Linear Velocity FMRLC Block Diagram

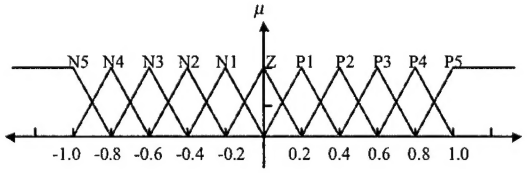

Figure 6: Normalized Input Membership Functions

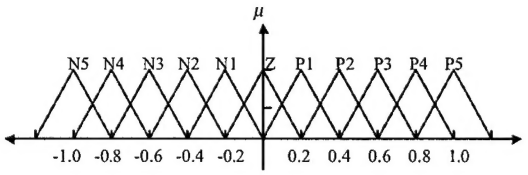

Figure 7: Normalized Output Membership Functions

Now that the inputs and the output of the fuzzy controller are defined, the rule-base for the inference mechanism must be defined. Typically, if little or nothing is known about the plant's characteristics, each rule's consequent is initialized to the linguistic variable "zero." This requires the controller to completely learn the system it is trying to control. By using the MAX architecture, an important conclusion about the plant's characteristics can be made. This conclusion is that increasing the term f_x should have the general characteristic of increasing v_x, and decreasing the term f_x should have the general characteristic of decreasing v_x. With this in mind, and using the membership function defined in Figures 6 and 7, the rule base for the fuzzy controller is initialized with the rules given in Table 1.

It is assumed in Table 1 that the pitch has no affect on the control of the AGV's linear velocity. This assumption is made initially because there is not enough information about the plant's characteristics to make a conclusion on how the pitch will affect the control of the AGV's linear velocity. Therefore, the controller must learn how to control the plant for different vehicle pitches.

Table 1: Linear Velocity Initial Rule-Base

Force		Desired Linear Velocity										
		N5	N4	N3	N2	N1	Z	P1	P2	P3	P4	P5
Pitch	N5	N5	N4	N3	N2	N1	Z	P1	P2	P3	P4	P5
	N4	N5	N4	N3	N2	N1	Z	P1	P2	P3	P4	P5
	N3	N5	N4	N3	N2	N1	Z	P1	P2	P3	P4	P5
	N2	N5	N4	N3	N2	N1	Z	P1	P2	P3	P4	P5
	N1	N5	N4	N3	N2	N1	Z	P1	P2	P3	P4	P5
	Z	N5	N4	N3	N2	N1	Z	P1	P2	P3	P4	P5
	P1	N5	N4	N3	N2	N1	Z	P1	P2	P3	P4	P5
	P2	N5	N4	N3	N2	N1	Z	P1	P2	P3	P4	P5
	P3	N5	N4	N3	N2	N1	Z	P1	P2	P3	P4	P5
	P4	N5	N4	N3	N2	N1	Z	P1	P2	P3	P4	P5
	P5	N5	N4	N3	N2	N1	Z	P1	P2	P3	P4	P5

The reference model takes the desired linear velocity as input and outputs an estimate of what the vehicle linear velocity should be. The model implemented here is a simple first order model. This was chosen for its simplicity where only one model variable needs to be determined, the time constant. This time constant is set to the system's average response time to various f_x commands.

The learning mechanism uses the linear velocity calculated by the reference model and the current AGV linear velocity to calculate an error, $e(kT)$ and change in error, $ce(kT)$. The error is scaled by the gain g_e and the change in error is scaled by g_{ce} in order to use the membership functions given in Figure 6 for fuzzification. These gains are determined by the maximum possible errors. Therefore g_e is set to $1/v_{desired}$ and g_{ce} is set to $T/v_{desired}$, where $v_{desired}$ is the desired tracking speed and T is the time interval. The rules used by the inference mechanism are given in Table 2. The conclusions of the rule-base are defuzzified using the COG and the membership function in Figure 7. And finally, the gain g_p is used to control how fast the system adapts and is left as a tuning parameter.

Table 2: Learning Mechanism Rule-Base

Change in process input		Change in error										
		N5	N4	N3	N2	N1	Z	P1	P2	P3	P4	P5
Error	N5	N5	N5	N5	N5	N5	N5	N4	N3	N2	N1	Z
	N4	N5	N5	N5	N5	N5	N4	N3	N2	N1	Z	P1
	N3	N5	N5	N5	N5	N4	N3	N2	N1	Z	P1	P2
	N2	N5	N5	N5	N4	N3	N2	N1	Z	P1	P2	P3
	N1	N5	N5	N4	N3	N2	N1	Z	P1	P2	P3	P4
	Z	N5	N4	N3	N2	N1	Z	P1	P2	P3	P4	P5
	P1	N4	N3	N2	N1	Z	P1	P2	P3	P4	P5	P5
	P2	N3	N2	N1	Z	P1	P2	P3	P4	P5	P5	P5
	P3	N2	N1	Z	P1	P2	P3	P4	P5	P5	P5	P5
	P4	N1	Z	P1	P2	P3	P4	P5	P5	P5	P5	P5
	P5	Z	P1	P2	P3	P4	P5	P5	P5	P5	P5	P5

4. ANGULAR VELOCITY FMRLC

The angular velocity FMRLC uses the block diagram given in Figure 8, which is very similar to the linear velocity FMRLC block diagram. Here the controller reference input, $\omega_{z,d}(kT)$, is the current desired angular velocity, and the input $v(kT)$ is the vehicle's current linear velocity. The linear velocity is chosen as an input since it is expected that more slip will occur between the vehicle tires and the ground at higher speeds, and therefore affect the vehicle's angular velocity. The gains, g_ω and g_v, are used again to normalize the inputs. The gain g_ω is chosen to be $1/\omega_{z,max}$, where $\omega_{z,max}$ is the vehicle's maximum angular velocity. Similarly, the gain g_v is chosen to be $1/v_{max}$ where v_{max} is the vehicle's maximum linear velocity. Again, the information required in order to calculate these gains are given either by the Vehicle Control Unit (VCU) configuration report or measured by the Position system (POS).

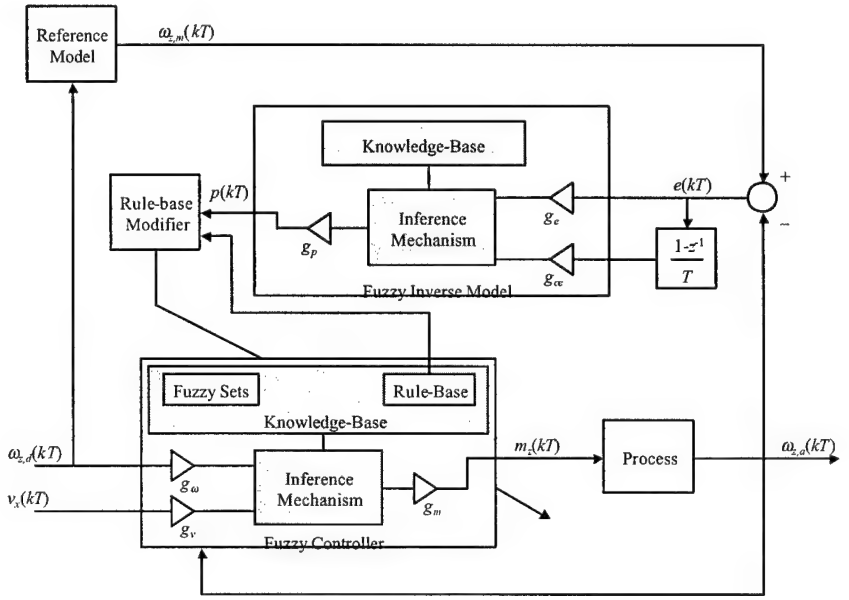

Figure 8: Angular Velocity FMRLC Block Diagram.

For vehicles with a nonzero minimum turning radius, $\omega_{z,max}$ turns out to be a function of the AGV's current linear velocity and its minimum turning radius:

$$\omega_{z,\max} = \frac{v_{current}}{r_{\min}}. \tag{5}$$

Note that when the current linear velocity is equal to zero, the gain g_ω for vehicles with a nonzero minimum turning radius is infinite. This is because the vehicle is not capable of turning unless the linear velocity is nonzero. Since it is impossible for the vehicle to turn unless the linear velocity is nonzero, the gain g_ω is set to zero if the linear velocity is zero. This is done so that the controller does not attempt to adapt for this case.

The controller's output in Figure 8, $m_z(kT)$, is the last term of the propulsive wrench. Using the output membership functions shown in Figure 7, the output of the inference mechanism is normalized. The gain g_m is used to scale this output to allow the controller to command the entire range of the term m_z. In the MAX architecture, the term m_z also has the range from –100 to 100 percent, and therefore the gain g_m is chosen to be 100.

Just as the MAX architecture provided information for the linear velocity controller, it also provides some information about the angular velocity in order to initialize the rule-base of its controller. It is expected that by increasing the term m_z in the propulsive wrench, the AGV's angular velocity will increase. And, by decreasing the term m_z in the propulsive wrench, the AGV's angular velocity will decrease. This is, of course, with the exception when the linear velocity is equal to zero as mentioned earlier. With this information, and using the membership functions defined in Figures 6 and 7, the rule-base for the angular velocity controller is initialized

Table 3: Angular Velocity Initial Rule-Base

Moment		Desired Angular Velocity										
		N5	N4	N3	N2	N1	Z	P1	P2	P3	P4	P5
Linear Vel.	N5	N5	N4	N3	N2	N1	Z	P1	P2	P3	P4	P5
	N4	N5	N4	N3	N2	N1	Z	P1	P2	P3	P4	P5
	N3	N5	N4	N3	N2	N1	Z	P1	P2	P3	P4	P5
	N2	N5	N4	N3	N2	N1	Z	P1	P2	P3	P4	P5
	N1	N5	N4	N3	N2	N1	Z	P1	P2	P3	P4	P5
	Z	N5	N4	N3	N2	N1	Z	P1	P2	P3	P4	P5
	P1	N5	N4	N3	N2	N1	Z	P1	P2	P3	P4	P5
	P2	N5	N4	N3	N2	N1	Z	P1	P2	P3	P4	P5
	P3	N5	N4	N3	N2	N1	Z	P1	P2	P3	P4	P5
	P4	N5	N4	N3	N2	N1	Z	P1	P2	P3	P4	P5
	P5	N5	N4	N3	N2	N1	Z	P1	P2	P3	P4	P5

with the rules given in Table 3.

It is assumed in Table 3 that the linear velocity has no affect on the control of the AGV's angular velocity. This assumption is made initially because there is not enough information about the plant's characteristics to make a conclusion on how the linear velocity will affect the control of the AGV's angular velocity. Therefore, the controller must learn how to control the plant for different linear velocities.

The reference model here takes the desired angular velocity as input and outputs an estimate of what the current vehicle angular velocity should be. The model implemented, like the linear velocity controller, is also a simple first order model. Again, this was chosen for its simplicity where only one model variable needs to be determined, the time constant. This time constant is set to the system's average response time to various m_z commands.

The learning mechanism uses the angular velocity calculated by the reference model and the current AGV angular velocity to calculate an error, $e(kT)$ and change in error, $ce(kT)$. The error is scaled by the gain g_e and the change in error is scaled by g_{ce} in order to use the

membership functions given in Figure 6 for fuzzification. These gains are determined again by the maximum possible errors. Therefore g_e is set to $1/\omega_{desired}$ and g_{ce} is set to $T/\omega_{desired}$, where $\omega_{desired}$ is the desired angular velocity and T is the time interval. The rules used by the inference mechanism are given in Table 2. The conclusions of the rule-base are defuzzified using the COG and the membership function in Figure 7. And finally, the gain g_p is used to control how fast the system adapts and is again left as a tuning parameter.

5. IMPLIMENTATION RESULTS

In addition to implementing the FMRLCs on the NTV for testing, they were implemented also on a K2A robot developed by Cybermotion, Inc., of Roanoke, Virginia (See Figure 9), and on an All-Purpose Remote Transport System (ARTS) (See Figure 10), which is a vehicle used by the United States Air Force Research Laboratory for research and design.

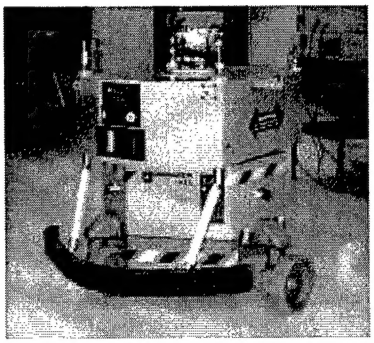

Figure 9: Cybermotion's K2A

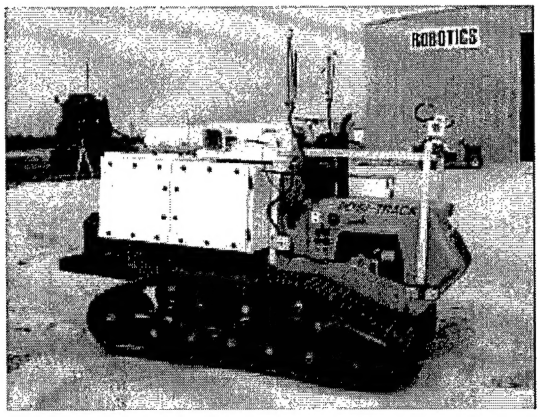

Figure 10: All-purpose Remote Transport System

Two different paths are used to test the two FMRLCs. A "U" shape path is used to test going from a straight section into a curve, and from a curve back into a straight section. And, a figure eight path is used to test going from a right curve into a left curve, and from a left curve into a right curve.

First, the FMRLCs were implemented on the NTV. The test paths were executed at speeds from 2 to 4 mps going forward, and it was also executed going backwards for the same range of speeds. Figures 11 and 12 show typical results of the NTV tracking a "U" shape path and a figure eight path, respectively. In both runs, the desired vehicle speed was set to 2 mps and the average vehicle position error was about 0.03 m and, the average velocity error was approximately 0.03 mps. Next, the FMRLCs were implemented on the K2A and tested. Typical results of these tests are shown in Figures 13 and 14. The desired velocity used for each of these tests was 0.15 mps. The average position error was 0.01 m and the average velocity error was 0.005 mps. Finally, the FMRLCs were implemented on the ARTS and tested using a "U" shape path and a figure eight path. The ARTS vehicle was already under tele-operated control. Its VCU was simply modified to accept wrench commands defined by MAX, and within one day it was navigating autonomously. Figures 15 and 16 show typical results of these tests, respectively, where the desired velocity was set to 1.4 mps.

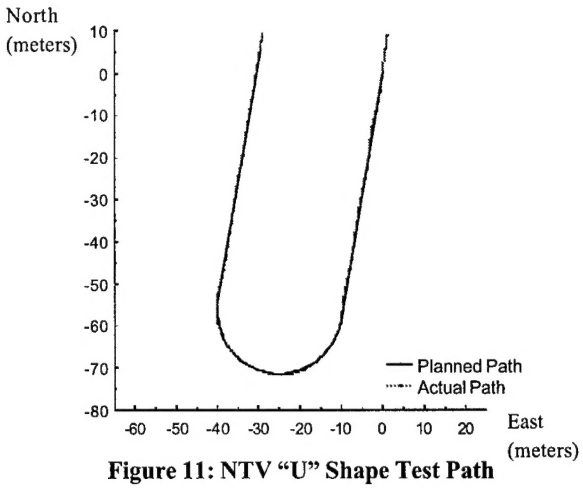

Figure 11: NTV "U" Shape Test Path

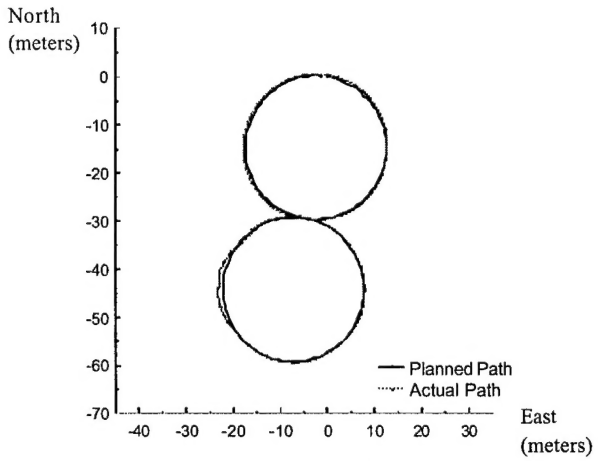

Figure 12: NTV Figure Eight Test Path

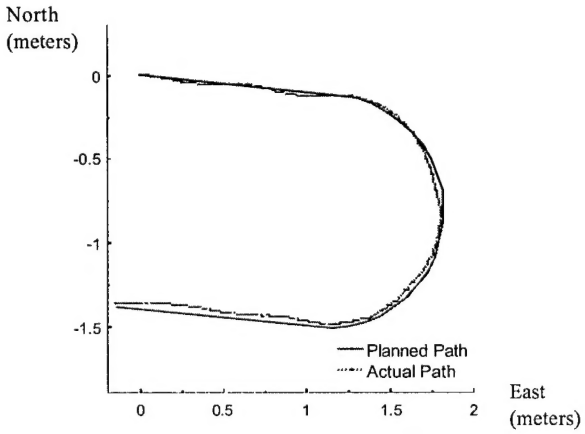

Figure 13: K2A "U" Shape Test Path

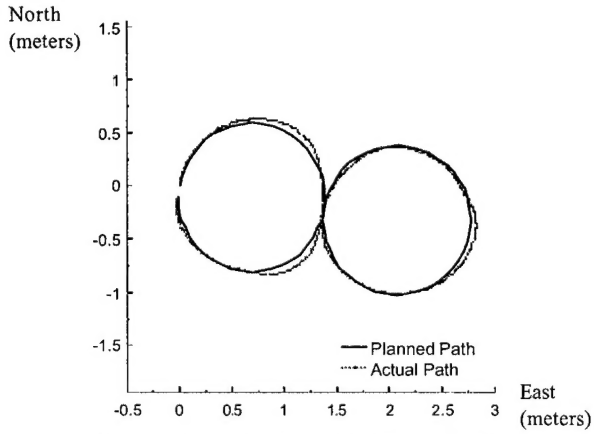

Figure 14: K2A Figure Eight Test Path

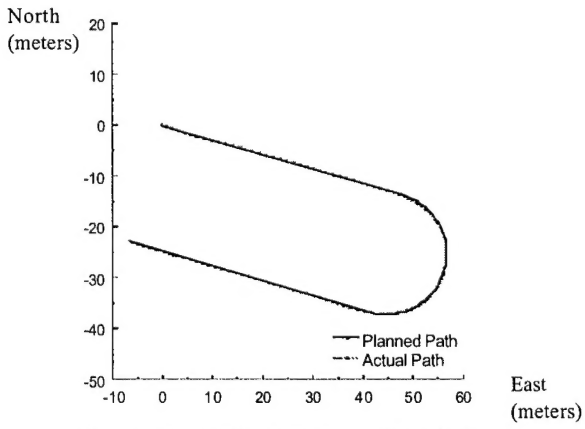

Figure 15: ARTS "U" Shape Test Path

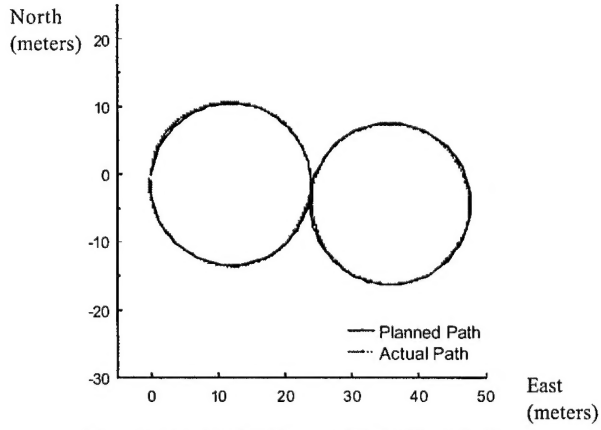

Figure 16: ARTS Figure Eight Test Path

6. CONCLUSIONS

In order to automate the navigation of a vehicle, controllers are required to track a desired velocity and a desired turning rate. A fuzzy model reference learning controller (FMRLC) was implemented to track the desired velocity, assuming that the user would set this tracking speed. This controller was designed from parameters of known vehicle characteristics such as the maximum speed and the maximum allowable pitch. A second FMRLC was implemented to track the desired vehicle turning rate. Again, this controller was designed from parameters of known vehicle characteristics such as the maximum turning rate and the maximum speed. Both controllers were implemented on the NTV and successfully tested.

By designing the controllers in term of known vehicle characteristics, transferring them to different vehicles was greatly simplified. In addition to implementing the FMRLCs on the NTV, they also were implemented and tested on a Cybermotion K2A and on an All-purpose Remote Transport System. In each case, the time required to implement the controllers required less than a day.

ACKNOWLEDMENTS

This work was performed under the sponsorship of the Air Force Research Laboratory, Tyndall Air Force Base Florida. Their continued support and encouragement are greatly appreciated.

REFREENCES

1. Rankin, A., and Crane, C., "Multi-Purpose Off-line Path Planning Based on an A* Search Algorithm," *Proceedings of the 1996 ASME Mechanisms Conference*, published on CD-ROM, 10 pages, Irvine, CA., 1996.

2. Rogers, R., Wit, J., Crane, C., Armstrong, D., "Integrated INU/DGPS for Autonomous Vehicle Navigation," *Proceedings of the 1996 IEEE Position Location and Navigation Symposium*, pp. 471-476, Atlanta, 1996.

3. Armstrong, D., Crane, C., Novick, D., Wit, J., English, R., Adsit, P., Shahady, D. "A Modular, Scalable, Architecture For Unmanned Vehicles," *Proceedings of the Association for Unmanned Vehicle Systems International (AUVSI) Unmanned Systems 2000 Conference*, Orlando, Florida, 2000.

4. Ball, Sir Robert Stawell, "A Treatise on the Theory of Screws," *Cambridge University Press*, Cambridge, United Kingdom, 1900.

5. Wit, J., Crane, C., Armstrong, D., and Duffy, J., "Autonomous Ground Vehicle Path Tracking," *Proceedings of the 32nd Southeastern Symposium on System Theory*, Tallahassee, Florida, 2000.

6. Passino, K.M. and Yurkovich, S., "Adaptive Fuzzy Control", *Fuzzy Control*, Addison Wesley Longman, Inc., pp. 303-315, Menlo Park, CA, 1998.